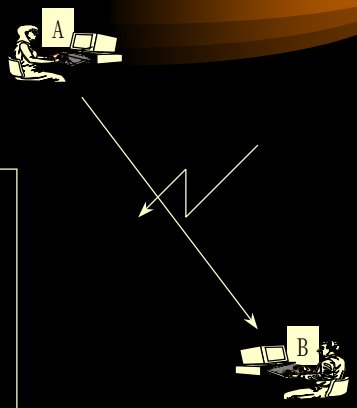
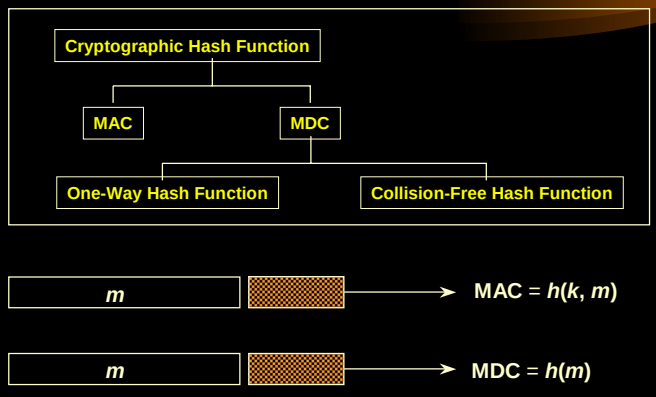
❑ 통신 네트워크 상의 위협요소

- ❑ Disclosure
- ❑ Traffic Analysis
- ❑ Masquerade
- ❑ Content Modification
- ❑ Sequence Modification
- ❑ Timing Modification
- ❑ Repudiation

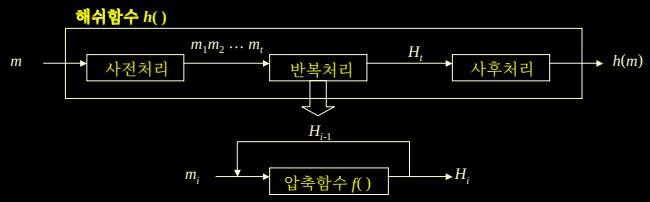


❑ 암호학적 해쉬함수의 분류

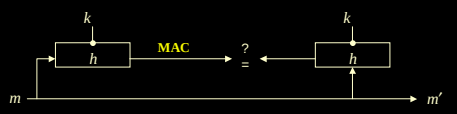


❑ 해쉬함수의 일반 모델

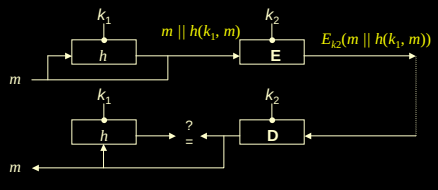
- e.g. MD4, MD5, SHA



❑ MAC (Message Authentication Code)



[그림 : **MAC**을 이용한 무결성 확인]



[그림 : 메시지의 기밀성을 동반한 **MAC**의 무결성 확인]

❑ MAC의 예

❑ MAC의 안전성

[정의] 안전하고 효율적인 MAC 해쉬함수 $h(\,)$는 다음 조건을 만족하는 함수이다.

① 함수 $h(\,)$는 공개된 함수이고 비밀키 k가 함수 계산에 사용된다.
② 변수 m의 길이에는 제한이 없고 $h(k, m)$의 길이는 l비트로 고정된다. ($l \geq 64$)
③ $h(\,)$, m 그리고 k가 주어졌을 때 $h(k, m)$의 계산은 용이하여야 한다.
④ 여러 쌍의 $\{m_i, h(k, m_i)\}$가 관측되었다고 할지라도 이를 이용하여 비밀키 k를 도출한다거나 또는 임의의 $m' (\neq m_i)$에 대해서 $h(k, m')$을 계산해 내는 것 역시 계산적으로 불가능(computationally infeasible)해야 한다.

암호이론과 보안 해쉬함수와 디지털 서명

❑ MAC의 안전성

다음 예는 마지막 조건을 만족하지 않는 MAC 해쉬함수와 그에 따른 능동적 암호공격자의 위조가 성공적인 경우를 보여 주고 있다. **t개의 64비트 블럭**들로 구성된 **메시지** $m = m_1 m_2 \ldots m_t$에 대한 MAC이 다음과 같이 계산되고, 그 메시지 m과 MAC이 능동적 공격자에 의해서 관측됐다고 가정하자.

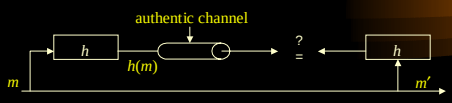
$$g(m) = m_1 \oplus m_2 \oplus \ldots\ldots \oplus m_t$$
$$\text{MAC} = h(k, m) = E_k(g(m))$$

능동적 암호공격자는 $m_1, m_2, \ldots, m_{t-1}$을 자신이 원하는 메시지 $m_1', m_2', \ldots, m_{t-1}'$으로 변조하고 m_t를 다음과 같은 m_t'으로 대체한다.

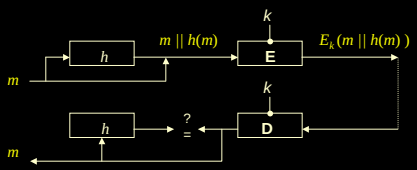
$$m_t' = m_1' \oplus m_2' \oplus \ldots\ldots \oplus m_{t-1}' \oplus g(m)$$

이때 $g(m)$과 $g(m') = m_1' \oplus m_2' \oplus \ldots\ldots \oplus m_{t-1}' \oplus \{ m_1' \oplus m_2' \oplus \ldots\ldots \oplus m_{t-1}' \oplus g(m) \}$은 동일하기 때문에 m을 m'으로 대체한다고 할지라도 관측된 MAC은 대체된 메시지에 대한 MAC과 일치하게 된다. 따라서, 능동적 암호공격자는 새로운 메시지 $m' = m_1' \,||\, m_2' \,||\, \ldots \,||\, m_t'$과 관측된 MAC을 함께 보내면 무결성 확인과정을 성공적으로 통과할 수 있게 된다.

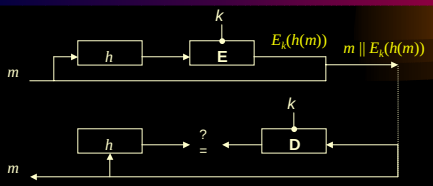
❑ MDC를 이용한 무결성 보장



❑ 메시지의 기밀성을 동반한 무결성 보장



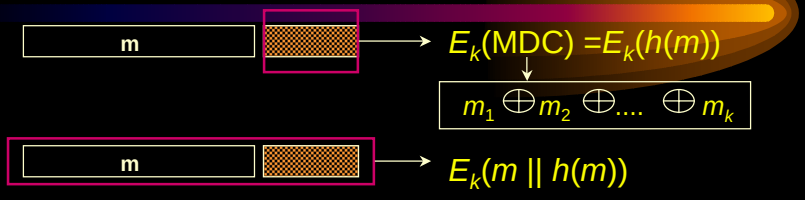
❑ MDC 암호화를 통한 무결성 확인



❑ MDC 해쉬함수의 안전성

[정의] 안전하고 효율적인 MDC 해쉬함수 $h(\,)$는 다음 조건을 만족하는 함수이다.
①함수 $h(\,)$는 공개된 함수이고 어떠한 비밀키도 함수 계산에 사용되지 않는다.
②변수 m의 길이에는 제한이 없고 $h(m)$의 길이는 l비트로 고정된다. ($l \geq 64$)
③$h(\,)$와 m이 주어졌을 때 $h(m)$의 계산은 용이하여야 한다.
④m과 $h(m)$이 주어졌을 때 $h(m') = h(m)$을 만족하는 m' $(\neq m)$을 찾는 것은 계산적으로 불가능해야 한다.

❑ MDC 해쉬함수의 안전성



$m_1\ m_2\ ...\ m_k\ m_{k+1}$, where m_{k+1} = MDC = $m_1 \oplus m_2 \oplus .... \oplus m_k$

DES-CBC => $c_1\ c_2\ ...\ c_k\ c_{k+1}$

$m_1 = IV \oplus D_k(c_1)$

$m_i = c_{i-1} \oplus D_k(c_i)$, $i = 2,3, ..., k$

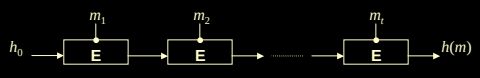
$m_{k+1} = c_t \oplus D_k(c_{k+1})$

❑ Examples of MDC Hash Function

- ❑ Rabin Hash Function

*Rabin*에 의해서 제안된 방식은 먼저 메시지를 블럭 암호에 적용될 입력의 크기와 동일한 블럭으로 나눈다. 메시지 m이 t개의 블럭 $m_1, m_2, ..., m_t$로 구성되고 초기화 벡터 IV를 이용하여 해쉬 값 $h(m)$은 다음과 같이 계산된다

$$h_0 = IV$$
$$h_i = E_{mi}(h_{i-1}),\ i = 1,2, ..., t$$
$$h(m) = h_t$$

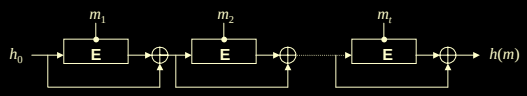


❑ Examples of MDC Hash Function

❑ Davies-Mejer Hash Function

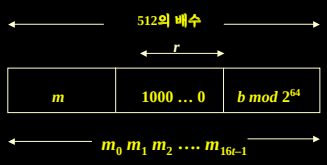
이 방식은 *Rabin* 방식을 개선한, 즉 '중간충돌' 공격으로부터 안전한 해쉬함수이다. 메시지 m이 t개의 블럭 $m_1, m_2, ..., m_t$로 구성된다면 해쉬 값 $h(m)$은 다음과 같이 계산된다.

$$h_0 = IV$$
$$h_i = E_{mi}(h_{i-1}) \oplus h_{i-1}, \quad i = 1,2, ..., t$$
$$h(m) = h_t$$



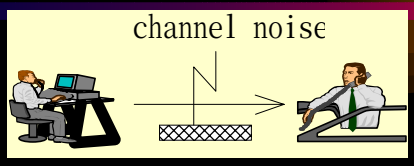
❑ 전용 소프트웨어 Hash Function

- ❑ MD4
- ❑ MD5
- ❑ SHA
- ❑ RIPEM-128
- ❑ RIPEM-160

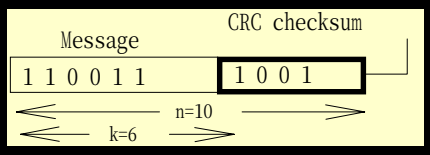


[그림 : MD5 메시지 사전처리]

❑ Error-Detecting Codes



- Generator Polynomial = $g(x) = x^4 + x^3 + 1$
- Message Polynomial = $m(x) = x^5 + x^4 + x + 1$
- How to generate CRC Checksum = $x^{n-k}\, m(x) \bmod g(x) = x^3 + 1$



❑ 필요성

- 송수신자간의 분쟁 (Dispute)
- 인증 (Authentication) 의 한계

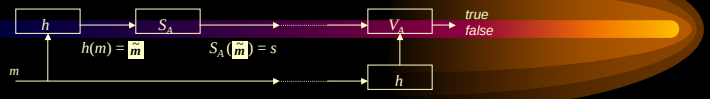
❑ 성질

- 서명자 확인가능
- 서명내용 인증가능
- 제 3 자에 의한 분쟁조정 가능

❑ 분류

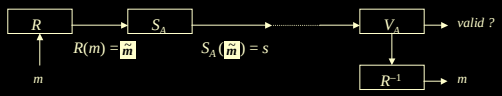
- 메시지 부가형 Digital Signature
- 메시지 복구형 Digital Signature

❑ 메시지 부가형 디지털 서명



- M : 서명자가 서명하려고 하는 메시지들의 집합
- S : 모든 가능한 디지털 서명의 집합
- 메시지 서명작업 $S_A : M \rightarrow S$
  - $\tilde{M}$: 재구성된 메시지들의 집합
  - $S_A : M \rightarrow \tilde{M} \rightarrow S$

❑ 메시지 복구형 디지털 서명



❑ RSA 디지털 서명

- $R(m) = \tilde{m} \in \{ 0, 1, \ldots, n-1 \}$
- $S_A(\tilde{m}) = \tilde{m}^d \bmod n = s$

- $\tilde{m} = s^e \bmod n$을 구한 후에
- $\tilde{m} \in R(M)$
- $m = R^{-1}(\tilde{m})$을 복구하게 된다.

RSA 서명 생성 : $S_A(R(m)) = S_A(\tilde{m}) = \tilde{m}^d \bmod n = s$
RSA 서명 확인 : $V_A(s) = s^d \bmod n = \tilde{m} \in R(M),\ R^{-1}(\tilde{m}) = m$

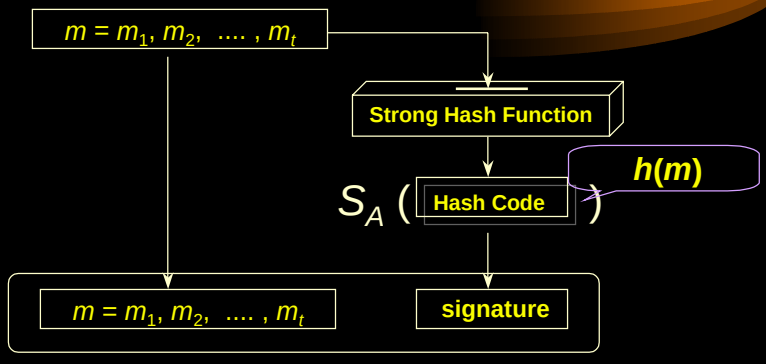
❑ ElGamal 디지털 서명

- 매우 큰 임의의 소수 p (소수 p는 현재 768비트 이상이 권고).
- 곱셈상의 군(group) Z_p^*의 생성자(generator) g 생성
- 난수 $x \in [1, p-2]$를 선정.
- 서명자의 공개키는 $\{ p, g, y = g^x \bmod p \}$, 개인키는 $\{ x \}$.

메시지 m에 대한 서명은 먼저 $gcd(\hat{y}, p-1) = 1$의 관계에 있는 난수 $\hat{y} \in [1, p-2]$을 선정하고, 메시지 m에 대한 해쉬함수 값 $c = h(m)$를 구한다. 메시지 m에 대한 디지털 서명 $\{ r, \hat{y} \}$은 다음의 계산을 통해서 생성된다.

$$\hat{y} = g^{\hat{x}} \bmod p$$
$$r = \hat{x}^{-1} \{ c - x \cdot \hat{y} \} \pmod{p-1}$$

❑ Practical Use of Digital Signature



❑ Collision-Free Hash Function

[정의] 충돌회피 해쉬함수 $h(\ )$는 다음 조건을 만족하는 함수이다.

①함수 $h(\ )$는 공개된 함수이고 어떠한 비밀키도 함수 계산에 사용되지 않는다.
②메시지 m의 길이에는 제한이 없고 $h(m)$의 길이는 l비트로 고정된다. ($l \geq 128$)③
$h(\ )$와 m이 주어졌을 때 $h(m)$의 계산은 용이하여야 한다.
④m과 $h(m)$이 주어졌을 때 $h(m') = h(m)$을 만족하는 m' $(\neq m)$을 찾는 것은 계산으로 불가능해야 한다.
⑤동일한 해쉬 값을 가지는 임의의 두 개의 서로 다른 메시지를 찾는 것은 계산적으로 불가능 하다는 의미에서 충돌회피의 성질을 가져야 한다.

❑ Birthday Attack

<1> 위조자는 $h(m) = h(m')$을 만족하는 두 개의 메시지 m과 m'을 생성한다. 메시지 m은 서명자에 의해 받아들여질 수 있는 내용이지만, 메시지 m'은 위조자에게 불법적인 이득을 초래하는 위조된 메시지이다.

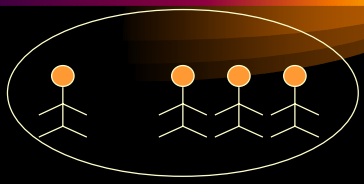
<2> 서명자는 위조자에 의해서 제시된 메시지 m에 대하여 서명을 한다.

<3> 서명자는 메시지 m'에 대해서는 서명을 하지 않으려 하지만 위조자는 결국 서명자의 서명 $S_A(h(m)) = S_A(h(m'))$을 얻게 된다.

Dear Anthony,

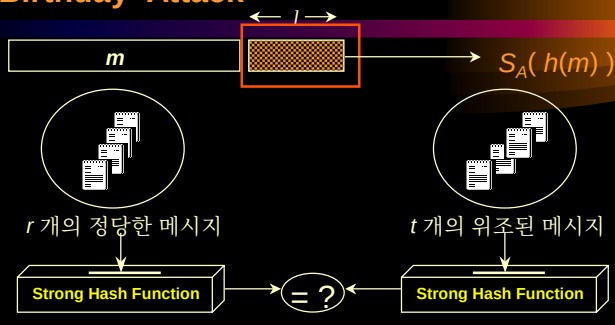
{ This letter is, I am willing } to introduce { you to, to you } { Mr., _ } Lee, the { new, newly appointed } {chief, senior } jewellery buyer for { you, the } Northern { European, Europe } { area, division }.

❑ Birthday Problem



- ❑ 어느 집단에서 생일이 같은 한 쌍이상의 학생이 존재할 확률(p)이 0.5이상이 되기 위한 학생수(r) ?
- ❑ $p = 1 - 1(1-1/n)(1-2/n) \ldots (1-(r-1)/n)$
- ❑ When n=365 and r >= 23, p >= 0.5

❑ Birthday Attack



- ❑ 비교가 실패할 확률 = 1 - 1/n when $n = 2^l$
- ❑ 동일한 해쉬 값을 가지는 메시지 한 쌍 이상이 발견될 확률

  $p = 1 - (1 - 1/n)^{r\,t} \sim 1 - e^{-r\,t\,l\,n}$
- ❑ $p \sim 0.63$ when $r = s = 2^{l/2}$